AFRL-IF-WP-TM-2003-1548

COPPER: COMPILER-CONTROLLED ON-DEMAND APPROACH TO POWER-EFFICIENT COMPUTING

Nikil Dutt, Rajesh Gupta, Alex Nicolau , and Alex Veidenbaum

University of California, Irvine
Department of Information and Computer Science
Irvine, CA 92697-3425

SEPTEMBER 2003

Final Report for 20 June 2000 – 09 January 2003

Approved for public release; distribution is unlimited.

STINFO FINAL REPORT

INFORMATION DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7334

NOTICE

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.

/s/ ________________________________
ANDREW W. HYATT, 2Lt, USAF
Program Monitor

/s/ ________________________________
ROBERT A. EHRET, Chief
Collaborative Simulation Technology & Applications Branch
Information Systems Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE ***(DD-MM-YY)*** September 2003	**2. REPORT TYPE** Final	**3. DATES COVERED** ***(From - To)*** 06/20/2000 – 01/09/2003

4. TITLE AND SUBTITLE

COPPER: COMPILER-CONTROLLED ON-DEMAND APPROACH TO POWER-EFFICIENT COMPUTING

5a. CONTRACT NUMBER
F33615-00-C-1632

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
69199F

6. AUTHOR(S)

Nikil Dutt, Rajesh Gupta, Alex Nicolau, and Alex Veidenbaum

5d. PROJECT NUMBER
ARPI

5e. TASK NUMBER
FS

5f. WORK UNIT NUMBER
A5

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of California, Irvine
Department of Information and Computer Science
Irvine, CA 92697-3425

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Information Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson AFB, OH 45433-7334

DARPA/IPTO
3701 Fairfax Drive
Arlington, VA 22203-1714

ONRRO San Diego
4520 Executive Drive, Suite 300
San Diego, CA 92121-3019

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/IFSD

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-IF-WP-TM-2003-1548

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
Report contains color.

14. ABSTRACT

The goal of this research was to build and demonstrate a capability in hardware (processors-memory) and software for management of power resources, and explore its trade-off against speed, accuracy and throughput requirements. The emphasis of this project was as much on utilization of the most power efficient architectural and software techniques as on their coordinated management across hardware and software levels. Through a coordinated management of power-control knobs from compiler to architectural and micro-architectural strategies we were able to achieve a range of power adaptation versus performance needs. However, instead of only leveraging on low power design approaches used in the embedded world and building on our own previous research on adaptive system software, we were also able to investigate power adaptation at the hardware and software levels. Our techniques collectively achieved a factor of 2-4x power/energy reduction.

15. SUBJECT TERMS
Compiler strategies for power management, data cache energy optimization, instruction cache energy optimization, dynamic energy management for a time-varying power profile

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT:**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** (Monitor)
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			2Lt Andrew W. Hyatt
Unclassified	Unclassified	Unclassified	SAR	20	**19b. TELEPHONE NUMBER** *(Include Area Code)* (937) 904-9162

Table of Contents

Executive Summary 1

Project Participants 3

Summary of Accomplishments 3

Instruction Cache Optimization 3

Data Caches Energy Optimization 5

Combined Savings from Multiple Techniques 6

Adapting Power Consumption Profile to Available System Power 7

Power PC Simulation Platforms 9

Design of Power-Aware Architectural Description Language (ADL) 10

Results for PAC/C Benchmarks 11

Technology Transfer 12

Selected COPPER Publications 13

DARPA Contract F33615-00-C-1632 Final Report

COPPER: Compiler-Controlled On-Demand Approach to Power-Efficient Computing

PIs: Nikil Dutt, Rajesh Gupta, Alex Nicolau , Alex Veidenbaum

Institution: Dept. of Information and Computer Science
University of California
Irvine, CA 92697-3425
Email {dutt, rgupta, nicolau, alexv} @ ics.uci.edu
Tel/fax: 949-824-4079
Project URL: **Project URL**: http://www.ics.uci.edu/~copper

Executive Summary

Objective: Adapt power consumption profile to application and system needs, minimize energy consumption.

The goal of this research was to build and demonstrate a capability in hardware (processors-memory) and software for management of power resources, and explore its tradeoff against speed, accuracy and throughput requirements. The emphasis of this project was as much on utilization of the most power efficient architectural and software techniques as on their coordinated management across hardware and software levels. Through a coordinated management of power-control knobs from compiler to architectural and micro-architectural strategies we were able to achieve a range of power adaptation versus performance needs. However, instead of only leveraging on low power design approaches used in the embedded world and building on our own previous research on adaptive system software, we were also able to investigate power adaptation at the hardware and software levels [4]. The ultimate goal was to automatically adapt power consumption and performance so that the same hardware could deliver high performance at the cost of increased power consumption, or bring power consumption to levels below those characterizing modern embedded systems at as minimal a cost to performance as possible. Automatically conforming to a time-varying available power profile was also sought. In order to achieve this goal, we took a synergistic approach to micro-architecture and compiler design. We simultaneously tackled both the static aspects of the problem as well as the dynamic (adaptive) aspects starting from a comprehensive study, to better understand, document and characterize the interactions between algorithms, compilers as well as hardware, with respect to power consumption.

Overall Goals of the Compiler-Controlled On-Demand Approach to Power-Efficient Computing (COPPER) project can thus be summarized as

1. Adapt power consumption profile to application and system needs

2. Minimize energy usage via architectural means whenever possible

Approach:

To understand the scope of power management at our disposal, we took a bottom-up view of the system ranging from underlying micro-architectural components to middleware and application software. Based on our exploration, we focussed on three key elements in order to achieve the above goals:

1. Architectural and Micro-architectural Power Smarts
    - “Power-aware” elements and their VLSI implementation
2. System Interfaces: Device, Compiler and OS
    - Compiler- and OS-level performance/power contracting and scheduling
    - ADL-driven architectural interfaces
3. Compiler Strategies for Power Management
    - Compiler-directed architectural “configuration”
        - generate “configuration code” embedded in the application
        - code “adapts” to new architectural organization at runtime
    - Power-use Estimation for Compiler Control
        - static analyses to select “optimal” configuration
        - static or dynamic prediction methods

In terms of micro-architectural components/modifications, we have identified the most promising architectural techniques for power-management that may be used with or without hardware enhancements.

Compiler control in identification and selection of adaptive architectural mechanisms as well as its coordinated execution by the runtime system is crucial to ensure machine usability and optimality of power/performance tradeoffs. To achieve this goal, we also explored semantic retention techniques that enable compilers to determine memory-specific application characteristics such as access patterns, memory footprint, detect array references that cause memory conflicts. An adaptive machine definition (ADL) model has been developed that would be used for power-aware simulator generation. For application-level power management for power aware computing communications (PAC/C) (specifically we worked on the unmanned arial vehicles[UAV] and NASA/JPL applications) we worked with the Integrated Management of Power Aware Computing and Communication Technologies (IMPACCT) and JPL (Jet Propulsion Laboratory) teams and have defined API’s for OS/real-time interaction.

Project Participants

Faculty PIs: Alex Nicolau, Nikil Dutt, Alex Veidenbaum, Rajesh Gupta

Students: Ana Azevedo, Ana-Maria Badulaescu, Radu Cornea, Paolo D'Alberto, Ilya Issenin, Ravindra Jejurikar, Cristiano Pereira, Weiyu Tang

Ana-Maria Badulaescu received her M.S. in June of 2001. Ana Azevedo received her PhD in Dec. 2002.

Summary of Accomplishments

The work performed in this project targeted both high-performance and embedded processors and fell into the following major areas:

1. Energy optimization of instruction caches
2. Energy optimization of data caches
3. Dynamic energy management for a time-varying available power profile
4. CAD support for architectural and energy modeling

The following sections summarize the major accomplishments in each area.

Instruction Cache Optimization

Instruction cache typically consumes 12-15% of the total processor energy. In this project several new cache designs have been proposed which can significantly reduce the energy consumption.

Predictive Instruction Filter Cache

A small and energy efficient filter cache can be placed between the CPU and the instruction cache to provide the instruction stream. There is, however, a loss of performance when instructions are not found in the filter cache. We proposed a prediction technique to reduce the performance loss associated with a filter cache in a high performance processor. It dynamically predicts whether the next fetch address will miss in the filter cache. In case a miss is predicted, the miss penalty is reduced by accessing the instruction cache directly. Experimental results show that the prediction technique reduces performance penalty by more than 91% and achieves an 82% energy-efficiency. Average instruction cache energy savings of 31% are achieved with a 1% performance degradation.

Decode Filter Cache for Embedded Processors

In embedded processors, instruction fetch and decode can consume up to 40% of processor power. Power savings can be achieved at either the fetch stage by storing instructions in a smaller filter cache or at the decode stage by caching the decoded instructions.

We proposed a small decode filter cache to provide decoded instruction stream. On a hit in the decode filter cache, the fetch from the instruction cache and the subsequent decode are eliminated, resulting in significant power savings in both instruction fetch and instruction decode. The efficiency of the decode filter cache is further improved by classifying the instructions into cacheable or non-cacheable, depending on the decoded widths. In addition, a prediction mechanism is used to reduce the decode filter cache miss penalty. Experimental results show 50% more power savings than an instruction filter cache and average 34% reduction in processor power with less than 1% performance degradation.

Integrating I-cache Way Predictor and Branch Target Buffer to Reduce Energy

In a set-associative cache, power savings can be achieved by accessing one cache way speculatively. There is performance degradation when the speculation is not correct. We proposed a Branch Target Buffer (BTB) design to reduce energy dissipation in set-associative instruction caches and to minimize performance loss. The functionality of a BTB is extended by caching way predictions in addition to branch target addresses. Way prediction and branch target prediction are done in parallel. Instruction cache energy savings are achieved by accessing only one cache way if the way prediction for a fetch is available. The best BTB configuration shows a 74% energy savings on average in a 4-way set-associative instruction cache and the performance degradation is only 0.1%. When the instruction cache energy and the BTB energy are considered together, the average energy-delay product reduction is 65%.

Power-Efficient Instruction Fetch Architecture for Super-Scalar Processors

An instruction cache of today's super-scalar processor delivers four or more 32b instructions every clock cycle and usually has a high degree of associativity. The required I-cache organization has a high energy consumption, somewhere between 10 to 20% of the total processor power. Not all of the simultaneously fetched instructions in a cache line are used by the processor due to branches. This research proposed a predictor-based instruction fetch mechanism. It determines which of the instructions in a given cache lines will be actually used and fetches only these useful instructions. It has no detrimental effect on performance. Using this approach results in a 17% reduction in the I-cache power consumption, on average, for an I-cache with a 16B line. For a 32B line the average savings are 33%. The additional power required by the predictor is very small.

Dynamic L0 Instruction Cache with History-based Prediction

A small Level 0 (L0) cache on top of a traditional L1 cache has the advantages of shorter access time and lower power consumption. The downside of a L0 cache is possible performance loss in case of cache misses. We analyzed the L0 instruction cache miss patterns and have proposed an effective L0 instruction cache management scheme through history-based prediction. For SPEC2000 benchmarks, the prediction hit rate is as high as 99% and the average hit rate is more than 93%. Compared to other L0 instruction cache management schemes, the new scheme eliminates more than 95% of the performance degradation in L0 caches while maintaining the energy advantage as shown by a lower energy-delay product.

Data Caches Energy Optimization

Data caches, although organized differently from instruction caches, consume about the same 15% percent of total energy. The number is even higher for embedded processors. However, fewer techniques exist for addressing this problem. Some of our work has been very successful in this area.

Data Cache Energy Reduction Through Way-Determination

Modern processors use data caches with higher and higher degrees of associativity in order to increase performance. A set-associative data cache consumes a significant fraction of the total energy budget, especially in embedded processors. We proposed a technique for reducing the D-cache energy consumption and showed its impact on energy consumption and performance. The technique utilizes cache line address locality to determine (rather than predict) the cache way prior to the cache access. It thus allows only the desired way to be accessed for both tags and data. The design has no impact on performance and, given that it does not have mis-prediction penalties, it does not introduce any new non-deterministic behavior in program execution.

The proposed mechanism is shown to reduce the average L1 data cache energy consumption in embedded processors running the MiBench benchmark suite for 8, 16 and 32-way set-associate caches by, respectively, an average of 66%, 72% and 76%. These results were obtained with simple scalar and watch simulators. The absolute energy consumption savings from this technique increase significantly with associativity.

The proposed mechanism is shown to reduce the average L1 data cache energy consumption for 2, 4 and 8-way set-associate caches in high-performance processors by 33%, 51% and 60%, respectively, for SPEC95 benchmarks.

Reducing Data Cache Energy Consumption via Cached Load/Store Queue

High-performance processors use a large set-associative L1 data cache with multiple ports. As clock speeds and size increase such a cache consumes a significant percentage of the total processor energy. We proposed a method of saving energy by reducing the number of data cache accesses. This is done by modifying the Load/Store Queue (LSQ) design to allow "caching" of previously accessed data values on both loads and stores after the corresponding memory access instruction has been committed. It is shown that a 32-entry modified LSQ design allows an average of 38.5% of the loads in the SpecINT95 benchmarks and 18.9% in the SpecFP95 benchmarks to get their data from the LSQ. The reduction in the number of L1 cache accesses results in up to a 40% reduction in the L1 data cache energy consumption and in an up to a 16% improvement in the energy-delay product while requiring almost no additional hardware or complex control logic.

Combined Savings from Multiple Techniques

We have proposed a number of techniques which are effective in reducing energy consumption in various parts of the CPU.   Many of these can be combined for greater total energy savings.

The way-determination technique for reducing the energy consumption for data cache does not affect the processor timing and it only implies changes to way the data cache is accessed controller. This implies that by applying the techniques to a design that contains an integrated I-cache way predictor and branch target buffer, the energy consumption savings enabled by the three designs are additive. For a 4-way cache system the savings are 51% for the data cache and 74% for the instruction cache.

Furthermore, combining this with the energy-efficient fetch predictor for the I-cache, an additional 33% of the I-cache energy dissipation can be eliminated resulting in an 82% I-cache average energy savings. Finally, for the D-cache it is also combinable with LSQ cacheing further reducing the energy consumption and resulting in approximately 70% D-cache energy savings. Assuming 12% energy consumption for both the I- and the D-caches, this leads to an approximately 18% total energy savings for the entire CPU.

If one takes into account The architectural techniques create in this project do not address the main component of the total energy consumption in the CPU, the clock distribution logic which consumes approximately 35% of the total. Caches consume 37% of the functional unit energy if clock is not taken into account. The total energy savings in this case are nearly 28% of the CPU functional unit energy.

Adapting Power Consumption Profile To Available System Power

This part of the project dealt with system operation under a time-varying power budget. The goal was t o adapt the application power requirements to available power dynamically. This was accomplished using a power profiler, a compiler/simulator tool for estimating individual code function power / energy usage. The run-time system included a power scheduler that, given available power input by the OS, scheduled a lower-power version of the code to execute. The underlying hardware mechanism is voltage and frequency scaling which is accomplished under compiler control.

Fig. 1 demonstrates the result of applying this technique using the software developed in this project to show conformance to available power. It clearly shows our technique to be effective in conforming to available maximum power. However, this technique had sometimes led to increased execution time. In order to remedy this we developed a different scheduler which used check-point information to meet timing deadlines. This work is described below.

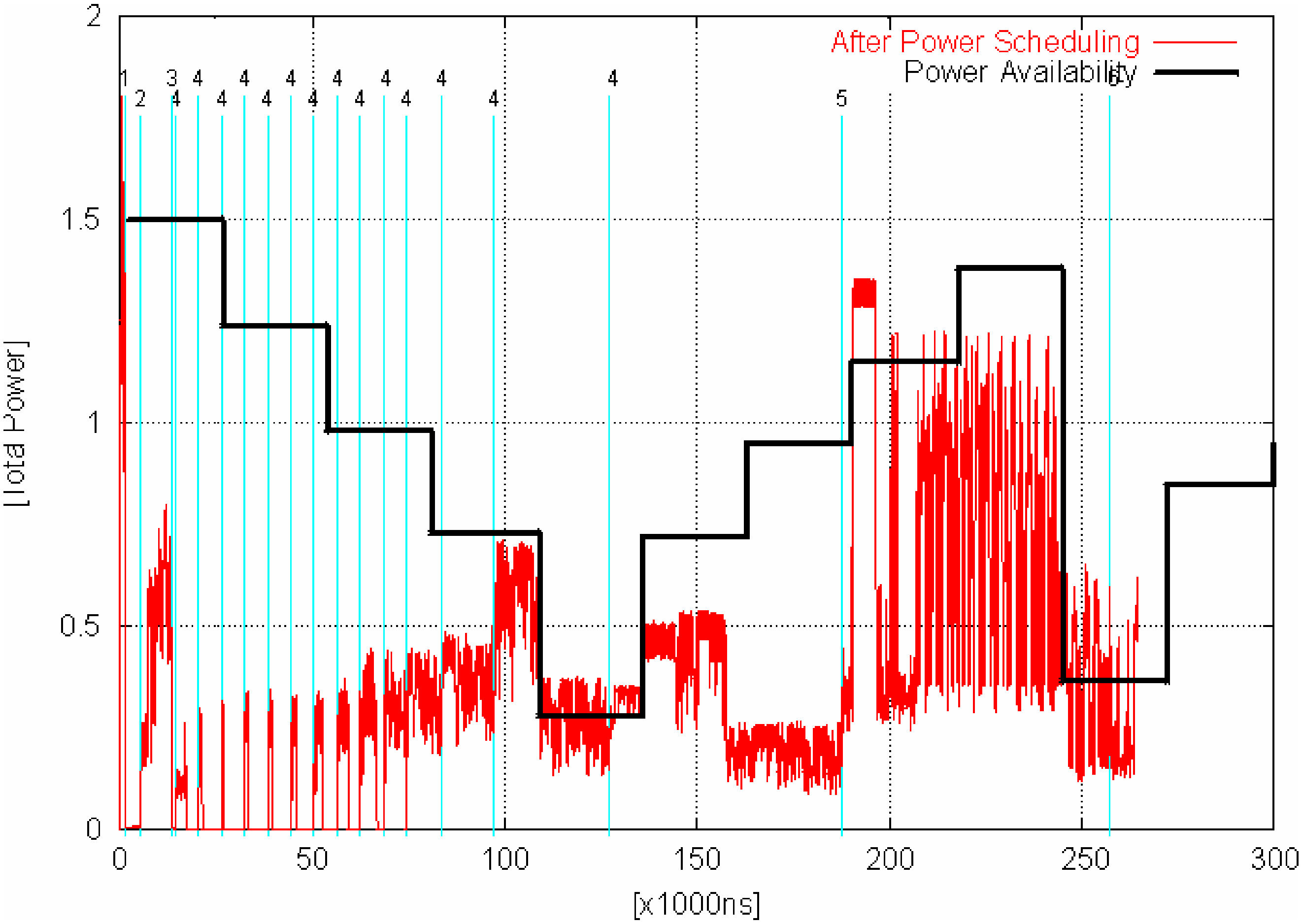

Figure. 1   Available Power Profile and Application Power Consumption

Profile-based Dynamic Voltage Scheduling using Program Checkpoints

Dynamic voltage scaling (DVS) is a known effective mechanism for reducing CPU energy consumption. While a lot of work has been done on inter-task scheduling algorithms to implement DVS under operating system control, new research challenges exist in intra-task DVS techniques under software and compiler control. Our research introduced a novel intra-task DVS technique under compiler control using program checkpoints. Checkpoints are generated at compile time and indicate places in the code where the processor speed and voltage should be re-calculated. Checkpoints also carry user-defined time constraints. Our technique handles multiple intra-task performance deadlines and modulates power consumption according to a run-time power budget. We experimented with two heuristics for adjusting the clock frequency and voltage. For the particular benchmarks studied, one heuristic yielded 63% more energy savings than the other. With the best of the heuristics we designed, our technique resulted in 82% energy savings over the execution of the program without employing DVS.

Power PC Simulation Platforms:

We have developped a new simulation package for Power PC based on the simulation capabilities of the IBM MET PowerPC and SimpleScalar simulators. And made these tools available to the PAC/C community.

METconsists of two parts: Aria – a workload trace generator and Turandot –an architectural simulator. Porting Turandot to Linux was relatively easy, since most
part of it is architecture independent (except for the code loader) and compiles on Linux without significant changes. On the other hand, Aria is heavily dependent on the underlying OS (system calls for code loading, specific memory addresses for loading the code, native execution of the instrumented workload). Tthus the approach we took on porting Aria involves replacing the native code execution by functional simulation from SimpleScalar for PowerPC. This also enables code loading to be done through SimpleScalar, and speeds up simulation time.

Design Of Power-Aware Architectural Description Language (ADL):

EXPRESSION is an Architecture Description Language (ADL) that can be used to rapidly generate a customized software toolkit (including a compiler, simulator, assembler and debugger). EXPRESSION tightly couples a programmer's instruction-set (or behavioral) view of the microarchitecture together with a computer architect's structural view (including details of the memory subsystem, pipelining, and timing information) into a succinct, formalized specification of the microarchitecture. The current instantiation of a software toolkit generates a compiler and a performance simulator that allows exploration of the specified microarchitecture using architectural parameters (affecting the instruction pipeline, the data path, the memory subsystem, and the execution logic), as well as through varying instruction set attributes (e.g., the addition or modification of specific instructions). However, the entire toolkit was not "power-aware". For the COPPER project, we have identified and implemented specific extensions the EXPRESSION ADL that facilitate the generation of a power-aware simulator. These can be classified into technology-dependent attributes (e.g., Voltage and Frequency), as well as a classification of power models of various microarchitectural components. We then verified the completeness and functionality of this power-aware ADL by using EXPRESSION descriptions as a front-end for generation of the Wattch architectural power simulator.

Results for PAC/C Benchmarks

1. **Frequency and Voltage Scaling through Program Checkpointing for UAV/SM Application**
   - **Time constraints for frame input (4 ms) + frame processing (12 ms)**
   - **Energy decreases by <u>38%</u> meeting both power and time constraints**

2. **IPC modulation for UAV/SM Application**
   - **Power constrained to half of the original average power consumption (4.3345)**
   - **Perfectly stabilizes power, but code runs 3 times slower and consumes <u>30%</u> more energy**

3. **History-based L0 I-cache management for UAV/SM Application**
   - **80% I-cache energy reduction due to high hit rate in the L0 cache**
   - **2% performance improvement due to quick instruction delivery by the L0 cache**

Technology Transfer

We have interacted with Intel, Motorola, IBM, and Tensilica in the course of this project. There was in the beginning a very active technology transfer with the PowerPC processor group at IBM; indeed we proposed to DARPA a joint research effort to utilize our software technology in a context of an IBM product which unfortunately this did not continue because it was not funded. The following is the list of our interactions and the description of the project results of interest to a given company.

1. Microcomputer Research Labs (MRL), Intel
    - □ POC: Dr. Utpal Banerjee (utpal.banerjee@intel.com)
        - ● Integration of Compiler Smarts into Intel Itanium platform

2. Tensilica
    - □ POC: Dr. Albert Wang (wang@tensilica.com)
        - ● Integration of Architectural Smarts into eXtensa

3. Motorola
    - □ POC: Pete Wilson (peter.wilson@motorola.com)
        - ● Integration of Microarchitectural smarts, ADL into embedded PPC platform

4. IBM
    - □ POC: Dr. Kemal Ebcioglu
        - ● Application of COPPER compiler technology for IBM PPC

Selected COPPER Publications

1. “Integrated I-cache Way Predictor and Branch Target Buffer to Reduce Energy Consumption”, W. Tang, A. Veidenbaum, A. Nicolau, R. Gupta, International Symposium on High Performance Computing, May 2002.
2. “Profile-based Dynamic Voltage Scheduling using Program Checkpoints in the COPPER Framework”, Ana Azevedo, Ilya Issenin, Radu Cornea, Rajesh Gupta, Nikil Dutt, Alex Veidenbaum, Alex Nicolau, Design Automation and Test in Europe, March 2002.
3. “An Efficient Compiler Technique for Code Size Reduction using Reduced Bit-width ISAs”, Ashok Halambi, Aviral Shrivastava, Partha Biswas, Nikil Dutt, Alex Nicolau, Design Automation and Test in Europe, March 2002.
4. “Memory System Connectivity Exploration”, Peter Grun, Nikil Dutt, Alex Nicolau, Design Automation and Test in Europe, March 2002.
5. “Automatic Verification of In-Order Execution in Microprocessors with Fragmented Pipelines and Multicycle Functional Units”, Prabhat Mishra, Hiroyuki Tomiyama, Nikil Dutt, and Alex Nicolau, Design Automation and Test in Europe, March 2002.
6. “Power Savings in Embedded Processors through Decode Filter Cache”, W. Tang, R. Gupta, A. Nicolau, Design Automation and Test in Europe, March 2002.
7. “Automatic Modeling and Validation of Pipeline Specifications driven by an Architecture Description Language”, Prabhat Mishra, Hiroyuki Tomiyama, Ashok Halambi, Peter Grun, Nikil Dutt, and Alex Nicolau, 7th Asia and South Pacific Design Automation Conference & 15th International Conference on VLSI Design, January 2002.
8. “Automatic Validation of Pipeline Specifications”, Prabhat Mishra, Nikil Dutt, and Alex Nicolau, Sixth IEEE International Workshop on High Level Design Validation and Test, November 2001.
9. “Architectural Adaptation for Power and Performance”, Weiyu Tang, Alexander V. Veidenbaum, Rajesh Gupta, International Conference on ASIC, October 2001.
10. “Architecture Description Language driven Design Space Exploration in the Presence of CoProcessors”, Prabhat Mishra, Frederic Rousseau, Nikil Dutt, and Alex Nicolau, Tenth Workshop on Synthesis And System Integration of Mixed Technologies, October 2001.

11. “Design of a Predictive Filter Cache for Energy Savings in High Performance Processor Architectures”, Weiyu Tang, Rajesh Gupta, Alexandru Nicolau, International Conference on Computer Design, September 2001.
12. “APEX: Access Pattern based Memory Architecture Exploration”, Peter Grun, Nikil Dutt, Alex Nicolau, 14th International Symposium on System Synthesis, September 2001.
13. “Functional Abstraction driven Design Space Exploration of Heterogeneous Programmable Architectures”, Prabhat Mishra, Nikil Dutt, and Alex Nicolau, 14th International Symposium on System Synthesis, October 2001
14. “Low Power Address Encoding using Self-Organizing Lists”, Mahesh Mamidipaka, Dan Hirschberg, Nikil Dutt, International Symposium on Low Power Electronics and Design, August 2001.